Eulerian Contact for Versatile Collision Processing

François Faure, Jérémie Allard, Matthieu Nesme

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Eulerian Contact for Versatile Collision Processing

François Faure — Jérémie Allard — Matthieu Nesme

N° 6203

Mai 2007

Thème NUM

ISSN 0249-6399 ISRN INRIA/RR--6203--FR+ENG

Eulerian Contact for Versatile Collision Processing

François Faure* , Jérémie Allard† , Matthieu Nesme*

Thème NUM — Systèmes numériques
Projets Evasion

Rapport de recherche n° 6203 — Mai 2007 — 20 pages

Abstract: We propose a new approach for collision modeling in physically based animation. Contrary to most current approaches, our method can be used with all physical models rather than any specific class. At each time step, the geometry of the bodies is *mapped* to a sparse regular Eulerian grid. Each grid point carries a mass, a velocity and a spatial density. This grid acts as a common mechanical layer where detection, modeling and reaction to collision take place, without any assumption about the internal dynamics of the bodies in contact. Mappings are then used to propagate penalty- and constraint-based reactions back to the bodies. We show that mappings can be easily set up for the most commonly used physical models. Our approach greatly simplifies the implementation of collision modeling since we only have to consider each body's individual mapping to the Eulerian grid, rather than numerous model pair-specific methods. Moreover, it allows us to design and reuse efficient collision response strategies independently of the physical models. We demonstrate our method with a variety of models including rigid bodies, deformable solids and fluids.

Key-words: collision, contact, simulation, physically based animation

* LJK/INRIA, France
† CIMIT Sim Group, Harvard Medical School, USA

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier (France)
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Contact Eulérien pour le Traitement Unifié des Collisions

Résumé : Nous présentons une nouvelle approche pour la modélisation du contact en animation par modèles physiques. Contrairement à la plupart des autres approches, notre méthode peut être appliquée à tous les modèles physiques plutôt qu'à une classe particulière. À chaque pas de temps, la géométrie des corps est plaquée sur une grille eulérienne par un *mapping*. Chaque point de la grille contient une masse, une vitesse et une densité spatiale. Cette grille joue le rôle d'une couche mécanique commune dans laquelle la détection, la modélisation et la réaction aux collision se produisent, sans connaissance de la dynamique interne des objets en contact. Le mapping est alors utilisé pour rétropropager les forces et contraintes de contacts vers les objets. Nous montrons que les mappings sont simples à construire pour la plupart des objets. Notre approche simplifie considérablement l'implémentation des contacts car il suffit de détermnier le mapping de chaque corps vers la grille eulérienne, plutôt que traiter une multitude de paires de modèles au cas-par-cas. De plus, elle permet de concevoir et réutiliser des stratégies efficaces de réponses aux collisions, indépendamment des modèles physiques. Nous en faisons la démonstration desur de nombreux modèles dont des solides rigides, des objets viscoélastiques et des fluides.

Mots-clés : collision, contact, simulation, animation par modèles physiques

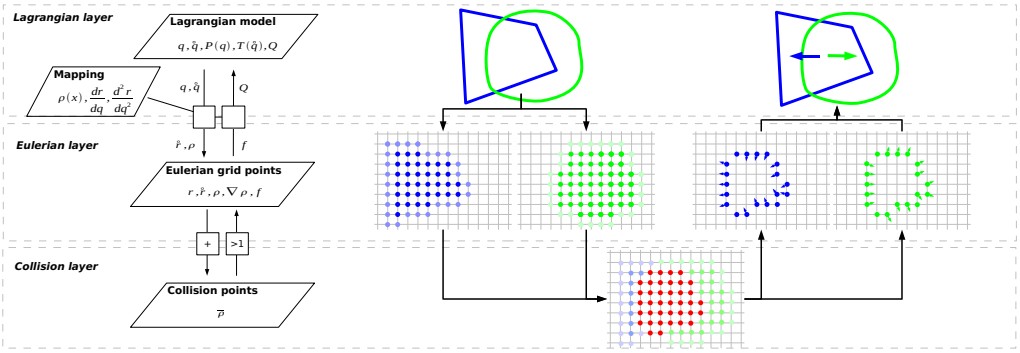

Figure 1: Overview of the method. The top layer contains arbitrary mechanical models. Mappings are used to project them to the Eulerian layer in the middle, where interactions take place. The bottom layer is the sum of densities used for collision detection.

1 Introduction

The field of collision processing has been thoroughly investigated for the last twenty years. However, none of the approaches proposed so far is both generic and efficient. Most methods are dedicated to a single type of contact such as triangle-based meshes. This makes the development of versatile physically based animation systems difficult involving the implementation and the run-time combination of numerous algorithms and techniques to deal with the variety of geometries (spheres, boxes, cylinders, lines or polygons) and mechanical behaviors (rigid, deformable, fluid). Using a new kind of geometric primitive requires implementing its intersection detection algorithm with each geometric primitive already used. This motivates us to build a totally new collision modeling method in which arbitrary mechanical models are mapped to a common collision model designed to detect and model contact efficiently.

1.1 Overview

Our model treats contact using a hierarchy of clearly separated layers, as illustrated in figure 1. The top layer contains arbitrary Lagrangian bodies governed by their own internal laws. Any type of physical body can be used, provided that a spatial description of its volume is available. We call this layer the *Lagrangian layer*. The lower layers allow interaction between bodies the top layer.

The second layer models the Cartesian space occupied by the bodies. At each time step, each body is sampled over a set of Eulerian points representing voxels with an associated filling ratio, which we call density. We call this layer the *Eulerian layer*. The physical interactions between the bodies are modeled at this level. Each body is sampled over the same set of points, resulting in a finite number of possible contact points.

The third layer, called the *collision layer*, is used for collision detection. It represents the sum of the volumetric images of the bodies. Interaction forces are applied at the collision points in the

direction of the gradient of density, and are then mapped to generalized forces applied to the bodies. This allows them to react to collisions.

Our contribution is to introduce a new contact modeling method based on Eulerian density grids, with the following properties:

- Any model can interact any other, provided a mapping from the body to the Eulerian layer is available.
- Density fields have the same versatility as distance fields for collision detection and modeling; however, they are much easier to resample at each time step because density is an additive, local property. Moreover, estimates of contact directions have better stability.
- Levels of detail are available for collision detection and modeling using coarse or fine versions of the density grids, allowing an easy trade off between geometric accuracy and speed.
- Collision response strategies, both penalty- and constraint-based, can be designed and re-used independently from the body models. This contrasts with traditional approaches which are targeted to specific models.

The remainder of this paper is organized as follows. Previous work is summarized in section 2. Section 3 discusses contact modeling and reaction more specifically. Section 4 shows how contact penalties and constraints are expressed in the Eulerian layer, and which operators are necessary to achieve the mechanical coupling with the bodies. Section 5 explains how to build a collision model for a variety of bodies at each time step. An architecture for versatile stable time integration is proposed in section 6. Results are presented in section 7, and a discussion is given in section 8.

2 Previous work

Collision detection and response techniques are well studied in the Computer Graphics community, and the contributions are too numerous to list here. Many different approaches have been proposed, depending on the type of objects in contact. While we provide a brief description below, please refer to [LG98, JTT01, TKH*04] for excellent surveys.

Rigid-body collision processing has been largely studied in the last two decades. While early work detected collisions between convex polyhedra [GJK88, GASF94, CLMP95] For non-convex polyhedra, various approaches using hierarchical bounding volumes were developed [BO04, GLM96, vdB97, KHM*98]. Alternatively, distance fields based methods have also been used [FSG03, FUF06]. Their advantage is that they do not have restrictions about the object topology and provide the penetration depth for collision response.

Deformable objects are not able to exploit as much precomputed data as rigid bodies, hence they require more complex collision detection algorithms. Many approaches use a dynamic hierarchy of bounding volumes [ZL03], and significant work has been done about the update of the hierarchy as body shape changes [LAM01, MKE03, JP04]. Other approaches use body-space hierarchies [VMT95, Pro97]. Recent work explores the use of volumetric data-structures, such

as spatial hashing [THM*03], GPU-accelerated voronoi diagrams [SGG*06], *Layered Depth Images* [HTG03]. Deformed distance fields [FL01, MAC04] have been used, however, they work for relatively small deformations. Most of the overhead of these techniques resides in updating the volumetric representation.

While contact response between rigid bodies is generally based on constraints [MW88, Hah88, Bar93, MC95], contact response for soft bodies can be based on penalty forces [BW98, VMT00] or constraints [Pro97]. [BFA02] combines penalties with constraints in a collision processing framework independent of the internal dynamics of simulated cloth.

Interactions with fluids are very challenging, due to the variety of representations and solvers used. While most solid simulations compute the positions and forces of a set of Lagrangian points, fluid solvers handle velocity and pressure fields defined over a fixed Eulerian grid [Sta99] or a set of Lagrangian particles [SAC*99, MCG03, PTB*03]. If the fluid is not present on the whole domain, its surface needs to be tracked using marker particles [FM96] or levelsets [FF01]. Recent approaches include combining particles and levelsets [EMF02, ELF05], using an octree representation [LGF04, HK05], and using a polygonal representation to enhance precision near the interface [BGOS06].

A simple approach to compute two-ways interactions between fluids and other objects is to consider them as fluids, either by adding particles at the surface of solids in contact with a particle-based fluid [GHD03, MST*04], or by considering rigid bodies as fluids with an additional rigidity constraint [CMT04]. More accurately, fluid-structure interactions are physically expressed by integrating the fluid pressure over the interface to provide the force applied to the solid, and using the solid's velocity as boundary conditions for the fluid [Ben92, GSLF05]. , or to accurately compute the coupling by remeshing around immersed objects [KFCO06, CGFO04].

While not strictly related to fluid-solid interactions, the recent work of Losasso *et al.* [LSSF06] presents a new, impressive method for handling multiple interacting fluids. Unfortunately it is not directly applicable to model interactions with other objects.

3 Background

In this section, we discuss the main approaches to model contact and apply reactions and show that our density-based contact model overcomes the drawbacks of the distance fields.

3.1 Contact modeling approaches

There are different classes of contact modeling approaches. The first is based on surface proximity. In this approach, contact is modeled as pairs of geometrical primitives (such as vertices, edges, and polygons) in close proximity and undergoing repulsive forces or constraints. However, primitives which cross the opposite surface are then repelled in the wrong direction. Consequently, a distance offset is typically needed to avoid intersections, and robustness requires adaptive time steps, continuous-time collision detection, and sophisticated mixes of elastic forces and hard constraints [BFA02].

The second class is based on penetration depth. In this approach, sample points on the surface of one object are tested against the volume of the other, as illustrated in the left of figure 2. Contact

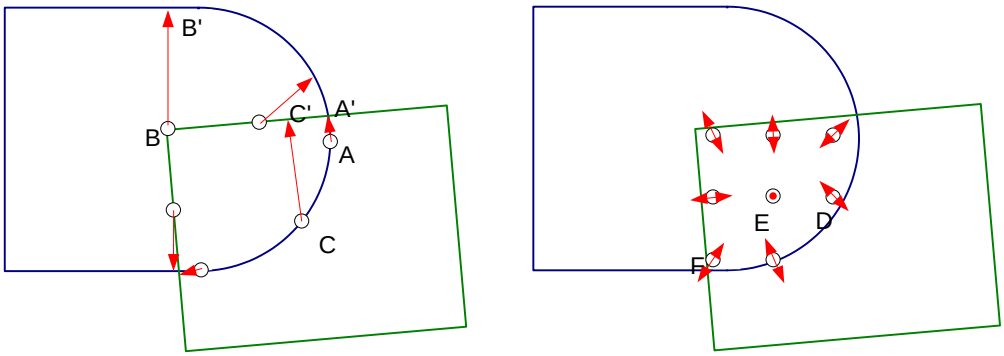

Figure 2: Contact modeling. Left: distance field. Right: density field.

is modeled as pairs of points and targets, such as (A,A') in figure 2. This approach is more robust because the primitives which cross the opposite surface are then repelled outward, which allows intersection. However, it can not be applied to pairs of lines and surfaces because they have no volume. Moreover, the directions of repulsion can be poorly approximated, like pair (B,B') in figure 2, or unstable, like pair (C,C') which depends on which side of the bisecting line of the square corner point C is located. The target points are the projections of the sample points to the closest primitive of the opposite surface. They are easily computed when simple volumes such as spheres or tetrahedras [THM*03] are used. Generally, finding targets involves the expensive computation and the storage of distance fields. Precomputation can be used for rigid bodies [GBF03], but sophisticated update strategies are necessary in the case of elastic bodies [FL01] and fluids [EMF02].

Recently, an alternative approach based on the minimization of the intersection volume has been presented by [VMT06]. They model the intersection volumes using the intersection lines and apply their technique to cloth. As far as we know, this approach has not been applied to volumetric objects due to the difficulty of efficiently modeling the intersection volumes. Our method overcomes this problem using an Eulerian density grid, as illustrated in the right of figure 2, and explained in section 4.

3.2 Contact reaction

Once contact is modeled, we have to compute how to move the points along the contact directions. The reaction methods basically fall in two categories, or a mix of both. Force-based methods apply elastic attraction between the contact points and their targets. Intersection occurs, and stiff springs are often necessary to keep it small enough. This requires either very small time steps to avoid instabilities, or the use of implicit time integration. Implicit integration uses the partial derivative of the forces with repect to the positions (*i.e.* stiffness). A linear equation system with a positive symmetric definite (PSD) matrix must be solved at each time step. Such equation systems have a unique solution and can be solved iteratively and efficiently using the conjugrate gradient algorithm [BW98]. The advantages of this approach are the stability, the numerical efficiency, and the good control of

the computation time. However, handling Coulomb friction is not straighforward. Moreover, implicit integration is possible only when contact forces are differentiable, which prohibits the use of distance fields due to the instability of the target directions (point *C*). Consequently, distance fields are generally used with contraint-based reaction methods.

Constraint-based methods aim to totally avoid intersections, by requiring the contact points to meet their targets. This generally results in singular systems which may have an infinite number of solutions as well as no solution at all, and sophisticated LCP solutions may be necessary [Bar94], with poor control of computation time. A popular alternative to solve such systems is to apply Gauss-Seidel iterations, where the constraints are processed sequentially until the residual falls under a given threshold, or a maximum number of computations is reached. The advantages of this approach are simplicity, versatility, easy Coulomb friction handling, and the control of the computation time. The drawbacks are a very slow converging rate when pressure waves propagate through numerous objects or through objects with highly different masses, as well as oscillations in case of conflicting constraints. Bridson et al [BFA02] present sophisticated methods to control proximities as well as strain using mixed force- and constraint-based strategies.

Force- and constraint-based reaction methods have different advantages and drawbacks, but they all rely on the target directions computed by the contact modeling method. Our new density-based contact modeling method has the same versatility as the distance fields, while overcoming its main drawbacks. Good, stable target directions are computed, allowing the implementation of robust elastic repulsion forces using implicit integration.

4 Eulerian contact

This section discusses the middle and bottom layers of figure 1. We show how to detect contact and how to set up the contact equations for force- and constraint-based responses.

4.1 Density fields

We call spatial density (or density, for short) the ratio of space locally occupied by matter. Density is a discontinuous function with value 1 inside a body, 0 outside, and 0.5 at the surface. However, we can get a continuous approximation of its value using finite differences in a regular grid. This creates as contact volume around the surface with the thickness of one step of the density grid. Near the surface, the density gradient gives an approximation of the normal, and bodies can repel each other along this direction. Though sharp edges can not be modeled exactly using density fields, we can get arbitrarily close approximations of them using appropriately fine sampling. Density fields can actually be seen as volumetric images of the bodies, and all the classical greymap processing techniques can be applied to control the level of detail. Figure 3 shows that smooth density fields can be modeled even with large downscaling factors. More sophisticated compression techniques such as wavelet bases could be used to increase computational efficiency. A key point of our mechanical framework is the very simple form of the collision detection criterion used at the bottom layer in figure 1: $\sum_i \rho_i > 1$ where index i ranges over all the bodies present at a given point. In the remainder

(a) (b) (c) (d)

Figure 3: Density Fields. The red line shows the 0.5 isovalue. (a): Original 256x256 image. (b),(c),(d): Same field reduced to the resolution of 16x16 and then rescaled to original resolution using subsampling, linear interpolation, and bi-cubic interpolation, respectively.

of this document, we only discuss pair-wise interactions. Interactions between more than two bodies at the same place are processed as pairs of interacting bodies.

The target directions are parallel to the gradient of density, as shown in sections 4.2 and 4.3. The gradient is null inside the collision volume (see point E in figure 2), and parallel to the normal near the surface (point D) or to a weighted sum of the normals (point F). The contact force is thus proportional to the external area of the contact volume, which is consistent with real-world physics. The density field can be sampled efficiently at each time step, as explained in section 5. Lines and surfaces are modeled as volumetric objects with thickness equal to the sampling step of the grid.

4.2 Contact penalties

When two bodies intersect, we can model an elastic penalty derived from a potential energy related to intersection. Using

$$\begin{array}{rcl} \rho_1+\rho_2>1 & : & \bar{\rho}=\rho_1+\rho_2-1 \\ else & : & \bar{\rho}=0 \end{array}$$

we define $E=\frac{1}{2}k\mathscr{V}\bar{\rho}^2$ as potential energy where $\mathscr{V}$ is the volume associated with contact point as a function of the grid resolution, and k is a user-defined parameter related to material properties. The following elastic forces at a given intersection point r can be derived from the potential

$$f_1=-\frac{\partial E}{\partial r}=-k\mathscr{V}\bar{\rho}\nabla\rho_1, \quad f_2=-\frac{\partial E}{\partial r}=-k\mathscr{V}\bar{\rho}\nabla\rho_2. \tag{1}$$

The density gradient $\nabla\rho=\frac{\partial\rho}{\partial r}$ is computed using finite differences in the regular grid. We use Eulerian point r as a contact point between the bodies. Newton's law $f_1=-f_2$ requires that $\nabla\rho_1=-\nabla\rho_2$ which is only true when the bodies are tangent. In practical simulations, the bodies intersect and the density gradients have different directions and intensities.

To maintain physical consistency, we have to average the forces computed using equations 1

$$f_1=-f_2=-\frac{k\mathscr{V}}{2}\bar{\rho}(\nabla\rho_1-\nabla\rho_2)\,. \tag{2}$$

A force orthogonal to the repulsion direction can be applied to model friction.

Note that these contact forces are only applied near the boundaries of the objects, where $\nabla\rho$ is not null. The net force is thus a function of the intersection area rather than intersection volume. When desired, it is possible to obtain volume-based penalties by decreasing the resolution of the contact grid.

Stiff penalties are necessary to avoid deep intersections. To maintain numerical stability, we have either to apply small time steps, or to perform implicit time integration [BW98]. The latter uses the stiffness operator, which relates a variation of force to a variation of body positions. The variations of force at points r and $r+\Delta r$ due to the displacement can be computed by differentiating the Eulerian force (eq. 2) as detailed in appendix A:

$$\frac{\partial f}{\partial r} = \frac{k\mathscr{V}}{4}\left((\nabla\rho_2 - \nabla\rho_1)(\nabla\rho_2 - \nabla\rho_1)^T + \bar{\rho}\left(\frac{\partial\nabla\rho_1}{\partial r} + \frac{\partial\nabla\rho_2}{\partial r}\right)\right) \tag{3}$$

which is a symmetric matrix.

4.3 Contact constraints

When constraint $\rho_1 + \rho_2 \leq 1$ is violated, we can solve it approximately using a first-order differentiation of the equation:

$$\frac{d(\rho_1+\rho_2)}{dr}\Delta r = -\bar{\rho}$$

which gives us the scalar equation

$$\begin{pmatrix} \nabla\rho_1^T & \nabla\rho_2^T \end{pmatrix}\begin{pmatrix} \Delta r_1 \\ \Delta r_2 \end{pmatrix} = -\bar{\rho}$$

where Δr_1 and Δr_2 are two displacement for a single point in the grid for the two bodies sampled at this point. This equation has 6 unknown scalars. We can constrain the displacement along the average gradient direction n given by

$$n = \frac{\nabla\rho_1 - \nabla\rho_2}{\|\nabla\rho_1 - \nabla\rho_2\|}$$

which reduces the equation to

$$\begin{pmatrix} \nabla\rho_1^T n & \nabla\rho_2^T n \end{pmatrix}\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = -\bar{\rho}$$

which has 2 unknown scalars. Newton's laws give us the additional equation $m_1 a_1 + m_2 a_2 = 0$ where m_1 and m_2 are the masses of the volume elements located at the grid point. This allows us to compute displacements at the grid particles. Continuous-time collision constraints can be modeled similarly. Note that due to finite differences, the computed displacement can not be larger than the Eulerian grid step. Resolving a deeper intersection would require several iterations.

5 Mappings

In the previous sections, we have shown that non-intersection penalties and constraints can be expressed using simple formulas based on ρ and $\nabla\rho$ at each contact point. In order to use these relations for the mechanical coupling of the intersecting bodies, we need to write these expressions as functions of the mechanical degrees of freedom (DOF) of the bodies. This section discusses the mapping between the top and middle layers in figure 1.

5.1 Mechanical requirements

Let q and $\dot{q}$ be the values and the time derivatives of the independent (also called *generalized*, or *Lagrangian*) DOF of the bodies. The associated forces are denoted using Q. In this section, we show which mathematical operators are necessary to map the DOF of the bodies to the Eulerian layer. The derivation of these operators in various cases is presented in section 5.

The interaction forces are computed at grid points, and we need to map these forces to generalized forces applied to the bodies. To do this in a physically consistent way, we have to ensure that the virtual power of the contact forces f is equal to the virtual power of their corresponding generalized forces Q. The virtual power of a contact force acting on a body is the dot product of the force by the virtual velocity $\dot{r}$ of a particle attached to the body and located at the contact point. The velocities at the contact points are given by

$$\dot{r} = \frac{\partial r}{\partial q}\dot{q}\,.$$

We thus need to compute Q so that $f^T \frac{\partial r}{\partial q}\dot{q} = Q^T\dot{q}$, where $\frac{\partial r}{\partial q}$ is the kinematic matrix relating the generalized velocities of the body to the velocity of the body particle located at the contact point. Since this relation must hold for any set $\dot{q}$ of generalized velocities, we get at each contact point

$$Q = \frac{\partial r}{\partial q}^T f\,.$$

When implicit time integration is needed, The stiffness matrix K of the contact force at each contact point is necessary. This operator encodes the variation of force in response to a small variation of q:

$$K_{ij} = \frac{\partial Q_i}{\partial q_j} = \frac{\partial}{\partial q_j}\left(\frac{\partial r}{\partial q_j}^T f\right) = \frac{\partial^2 r}{\partial q_j^2}^T f + \frac{\partial r}{\partial q_j}^T \frac{\partial f}{\partial r}\frac{\partial r}{\partial q_j}$$

where $\frac{\partial f}{\partial r}$ is given in equation 3. The kinematic operators $\frac{\partial r}{\partial q_j}$ and $\frac{\partial^2 r}{\partial q_j^2}$ make the connection between the contact layer and the bodies.

The application of hard constraints requires the computation of dq given dr. Since the velocity field at a given point can depend on an arbitrary number of mechanical variables, the pseudo-inverse of $\frac{\partial r}{\partial q_i}$ is required. The same is used to apply a velocity change. This allows the application of

instantaneous displacement or velocity changes, as commonly used in constraint-based methods. As usual, conflicting constraints can be solved iteratively using Gauss-Seidel approaches.

In summary, the necessary operators for the bodies to interact in the Eulerian layer are only $\rho(r)$, $\frac{\partial r}{\partial q}$ and $\frac{\partial^2 r}{\partial q^2}$. The first is the simplest geometric definition of a volume, while the second is the kinematic matrix relating particle velocities $\dot{r}$ to the generalized velocities $\dot{q}$, and the third encodes the non-linearity of $r(q)$. The other operators can be deduced using finite differences in the regular grid. In the remainder of this section, we explain how to build these operators for the most commonly used models.

5.2 Simple geometric primitives

Simple geometric primitives such as points, lines, triangles, tetrahedra and spheres can be sampled directly using techniques inspired from classical graphics rasterization. The sampling algorithm loops over the target points to evaluate the density. Points are sampled directly in the grid using antialiasing techniques. Lines can be sampled using a 3D variant of Bresenham's algorithm. Similarly, triangles and tetrahedra can be sampled by looping over the regularly spaced target points r using the appropriate variants of the traditional polygon-filling algorithms. Density $\rho(r)$ may be binary or real. In a translating sphere, each particle moves with the center and the mapping is the identity. In the other primitives, the velocities $\dot{r}$ are linearly interpolated like colors in graphical rendering. The interpolation coefficients are the non-null entries of the kinematic matrix $\frac{\partial r}{\partial q}$. The second-order $\frac{\partial^2 r}{\partial q^2}$ is null due to linearity.

5.3 Rigid bodies

When a shape is rigidly attached to a moving reference frame, the kinematic matrix encodes the standard rigid body velocity field $\dot{r} = \dot{o} + \omega \times (r - o)$, where o is the origin of the reference frame and ω is the angular velocity associated with a given kinematic variable. Its derivative $\frac{\partial^2 r}{\partial q^2} = \omega \times (\omega \times (r - o))$ encodes the curvature of the trajectories. Density can be evaluated by projecting back the Eulerian points to a precomputed distance or density field, or by projecting raw data to the Eulerian grid as explained in the next section.

5.4 Complex shapes

In the case of volumes defined by raw data such as medical images, or embedded in nonlinear deformation fields, we propose to use an additional layer, as illustrated in figure 4. We call it the volume layer. It represents the volume occupied by the body, using sampling particles attached to the body, with a volume assigned to each of them. The particles of this layer are not independent, since their positions and velocities $q', \dot{q}'$ are entirely defined by the particles of the top layer. The designer of the physical model is responsible for implementing the mapping to the volume layer. Fortunately, implementing the mapping associated with a complex-shaped physical body is as simple as depicting it as a set of spheres with centers, velocities (along with the kinematic operators discussed in section 5.1), radii and volumetric masses.

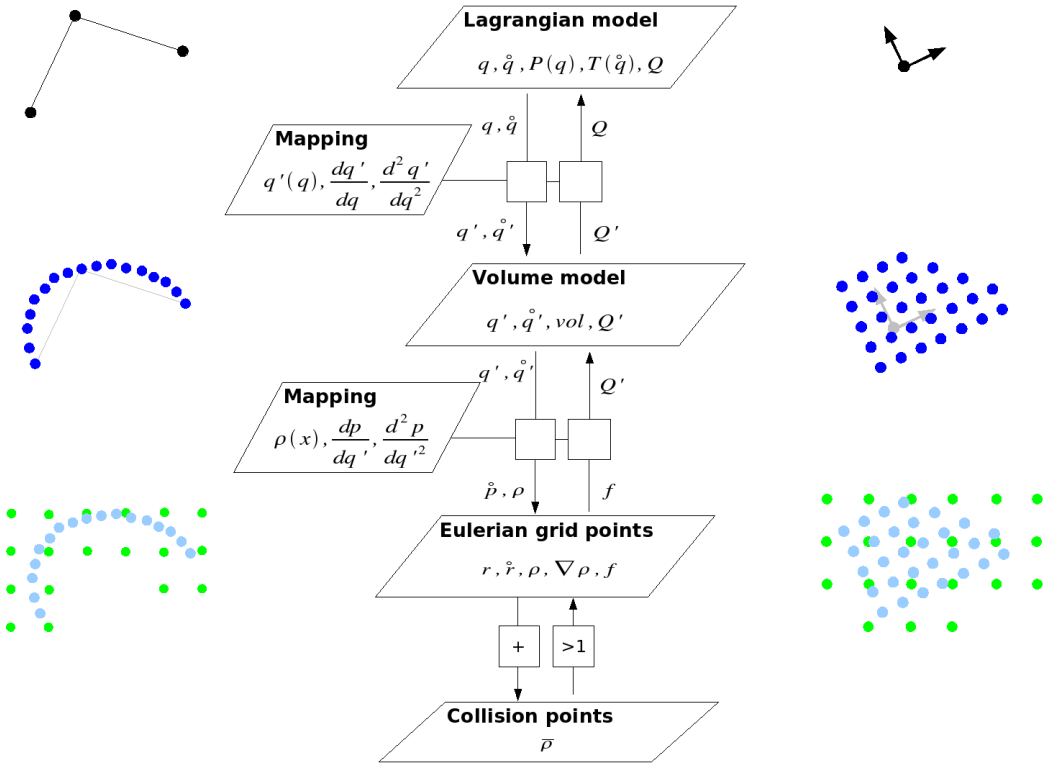

Figure 4: The volume layer inserted between the Lagrangian and Eulerian layers. Left: deformable body with nonlinear deformation, right: rigid body with raw volume data.

However, the resulting contact behavior is different from what can be obtained using traditional sphere-based contact models, because in our framework the contact spheres are then mapped to the Eulerian layer. This results in a discrete number of possible contact points, and much smoother contact surfaces, as shown in the accompanying video.

The mapping from the volume layer to the Eulerian layer is computed automatically as follows. Each particle distributes its density to the closest points in the Eulerian grid, and we average the contributions of several particles in the same Eulerian voxel. Let us consider a moving volume particle with position q'_i, velocity $\dot{q}'_i$ and mass m_i. The masses are used to compute physically consistent linear mapping coefficients. They can be straightforwardly deduced from the volume and the volumetric mass of the body. Physical consistency is achieved by distributing the mass and momentum of this volume particle to the corners of the Eulerian grid cell which contains it using its barycentric coordinates. The net mass and momentum distributed to an Eulerian grid point are the sum of the contributions of all the volume particles contained in its neighboring cells:

$$m_j = \sum_{i \in \text{body particles}} w_{ij} m_i \,, \tag{4}$$

$$(m\dot{r})_j = \sum_{i \in \text{body particles}} w_{ij} m_i \dot{q}'_i \tag{5}$$

where w_{ij} is the barycentric coordinate of volume particle i with respect to Eulerian point j. The density associated with a volume particle is its volume divided by the volume of the Eulerian cell. Once the mass m_j and momentum $(m\dot{r})_j$ associated with an Eulerian grid point are computed, its associated velocity and kinematic mapping coefficients are

$$\dot{r}_j = \frac{(m\dot{r})_j}{m_j} \quad , \quad \frac{\partial r_j}{\partial q'_i} = \frac{m_i}{m_j} \,, \tag{6}$$

The operator $\frac{\partial^2 r}{\partial q^2}$ is null due to linearity. Note that the radius of influence of the particles in the volume layer can be larger than one cell of the Eulerian grid. In this case, the density, the mass and the velocities are distributed over more than one cell.

The volume layer can be used to depict the surface of meshless deformable bodies. In this model, radial basis functions are attached to moving particles, the weighted sum of the particle displacements defines the kinematic field of the body, which is used to model the internal deformations, which is in turn used to compute appropriate response forces. This is a special case where the degrees of freedom q' of the volume layer are the same as the independent degrees of freedom q.

6 Implementation

While the fundamental equations have been discussed, implementing them efficiently requires several important considerations, detailed bellow.

While mathematically we can consider computing the density of all objects everywhere, in order to obtain good performances it is important to compute it only when necessary. In order to achieve

this we construct a hierarchical representation of the Eulerian grid. Whenever possible, only the root level of the hierarchy is computed, and finer levels are created on demand when more than one object is present. This strategy prunes the computation of most densities, except close to potential contacts. However in some cases the object is able to rasterize itself very efficiently, such as fluids internally computing a levelset, in which case it is more efficient to construct the hierarchy bottom-up.

In the previous sections, we have shown how potentially all physical models can exert interaction forces on each other through our Eulerian contact layer. We can thus consider the interactions as given external forces and animate each body independently using its dedicated differential equation solver. We have also designed an implicit time integration solver able to animate the interacting bodies when stiff forces and large time steps are applied. Consistently with our contact processing approach, the solver has no direct access to the internal degrees of freedom of the bodies. Each body stores internally its own state and auxiliary vectors including position, velocity, force and displacements. The solver uses the conjugate gradient algorithm, which triggers force computation and vector operations without addressing the vectors directly. The only feedback from the bodies to the solver is the value of dot products between auxiliary vectors. The computation of the net force applied to each body as a function of its positions and velocities is done recursively through the mappings. Stiffness computation is detailed in appendix B.

This approach makes no assumption about the type of degrees of freedom of the interacting bodies. One could be a viscoelastic body while the other could be a hierarchy of articulated rigid bodies, provided that the interaction forces are applied consistently. The method is reusable for all available physical bodies. Moreover, simple constraints such as maintaining an independent particle at a fixed place are internally applied by the bodies as filters. Notice that this flexible solver is not specialized to our contact framework and can be applied to traditional systems of bodies in contact. It allows us to straightforwardly apply [BW98]'s implicit time integration using a filtered conjugate gradient solution to our layered body framework, and we have been using it successfully.

We have also designed a generic constraint solver using the formulas given in section 4.3. Constraints are processed one after another, and each correction is immediately propagated, in a Gauss-Seidel manner. In our experiments, the stiff penalty approach gives more stable results. However, we plan to use it in future work to apply Coulomb friction using the robust method proposed by [BFA02].

7 Results

Our new framework allows us to animate scenes including arbitrary physical models. We implemented and tested many algorithms, including rigid bodies, deformable objects (using either mass-springs or FEM), an Eulerian grid-based fluid solver, as well as a Lagrangian SPH fluid.

To test the robustness of our method, we designed several simulations involving up to three different types of objects. Each algorithm is only completed by the mappings to and from the Eulerian density grid, and has no knowledge nor special codes to handle interactions to another specific type of objects. Despite this lack of specialized interactions, we were able to reproduce previous experiments, such as collisions between a fluid and a thin cloth [GSLF05] (figure 5).

While interactions between fluids, rigids and deformables have previously been demonstrated, coupling all three in a single simulation has not yet been clearly demonstrated. This is the motivation

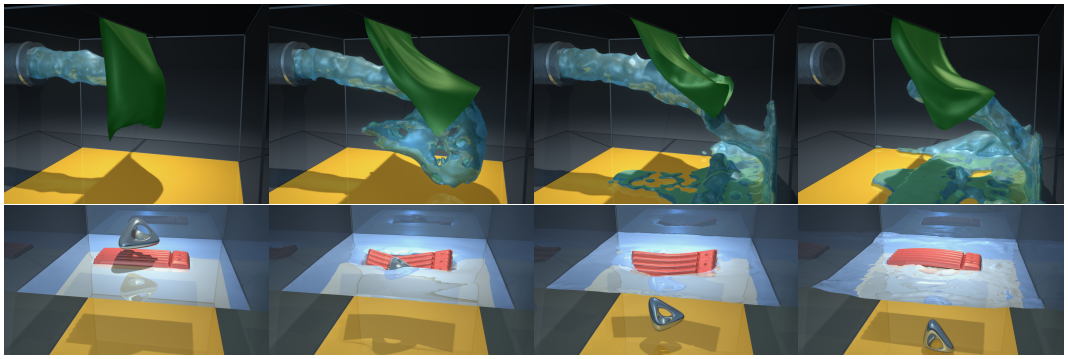

Figure 5: Top: contacts involving an SPH-based fluid and a thin deformable cloth. Down: Interactions between an Eulerian grid-based fluid, a co-rotational FEM deformable model, and a rigid body.

behind the animation shown in figure 5(down), involving a light poolrack floating over a fluid and crushed by an heavy rigid object falling to the ground. In this case the fluid solver is using a very different integration method than the rest of the simulation (semi-lagrangian advection, then pressure projection), hence the fluid is not included in the implicit integration system and interacts with the other bodies using forces integrated explicitly. While this requires a rather small timestep (of the order of $1/100^{th}$ second), it is nonetheless able to produce correct interactions. Expressing forces interacting with the fluid implicitly, as shown by Chenatez *et al.* [CGFO04], would allow for much larger timesteps.

The previously mentioned simulations were executed on a single processor at speeds varying between a few frames per seconds to less than 5 seconds per frame. More importantly, this speed was not significantly slower than each individual simulated object taken individually. In many cases, the fluid solver is the bottleneck. The scene shown in the second picture in the first page includes rigid and deformable volumetric bodies, as well as pieces of cloth. The total amount of particles is more than 21,000. As shown in 6, the collision detection and modeling takes less than 25% of the computation time, even in this complex stacking configuration.

8 Discussion

Our framework meets the objective of allowing contact between arbitrary physical models. Contrary to all previous work, our approach allows contact interactions without any assumption about object geometry (e.g. mesh, implicit function) or constitutive laws (e.g. rigid, visco-elastic, fluid). We demonstrate it using rigid, visco-elastic and fluid bodies with mesh-based and implicit geometries. The differentiability of the density-based contact forces allows the use of implicit time integration, resulting in stable simulations. Using appropriate penalty stiffness, contact errors are easily smaller than the size of one density voxel. Even in the complex case of interaction between fluid

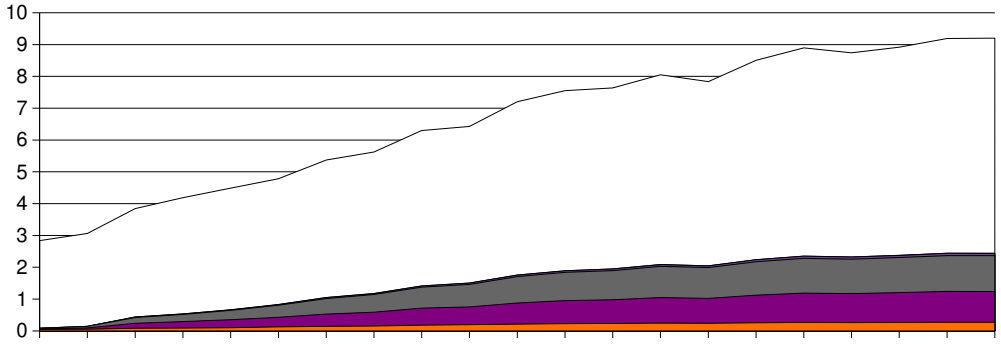

Figure 6: Performance measurements. The white upper area represents the time devoted to the computation and integration of internal and repulsion forces. The rest corresponds to collision detection and modeling.

and deformable bodies, the important phenomena such as buoyancy and the meeting of boundary conditions are present.

Our method is more efficient than distance fields. Density fields are computed much more efficiently because density is a local, additive property. This allows us to simulate deformable objects, while methods based on distance fields are generally limited to rigid bodies with precomputed distance fields, or, in the case of fluid simulation, involve complex interface tracking methods.

We believe that the computations in the Eulerian grid are very well suited to the hardware accelerations available with modern graphics architectures.

The limitations of our method are mainly due to density rasterization. The geometrical accuracy of our method directly depends on the resolution of the grid. Nonetheless, as shown in the first accompanying video, a very coarse grid gives surprisingly good results, and a finer grid makes intersections unnoticeable. Aliasing artifacts are hardly noticeable due to smooth density mapping. Note that in case of deformable bodies, contact precision is typically less of an issue, allowing a trade-off between accuracy and speed when needed.

Our current implementation has the limitation that the grid resolution must fit the smallest object in the scene. This results in non-interactive frame rates when thin objects such as cloth are involved. In future work, we plan to use hierarchical grids, and interpolate the coarse grids locally near the contacts.

In future work, there are a number of issues that we would like to address. Self-collision could be handled by partitioning an object's shape in several complementary sub-shapes tagged with different identifiers but referring to the same object. When a solid object is wholly contained within another solid object, the object inside will not be repelled because the forces acting on its surface will cancel each other. To handle this, time continuous collision detection could be used. Locally, we can estimate the time derivative of the density based on density gradients and velocities. When distant objects with high velocities are present, we could fill the voxels swept during the whole estimated

trajectory rather than only the current location, similarly with the expanded bounding boxes used by some authors.

Our approach does not compete with specialized methods when only one kind of object is involved. The benefit of our method is to allow any kinds of physical bodies to interact.

A Contact stiffness

When a body moves, its density field moves accordingly. We start from equation 2 and consider a small displacement Δr of body 1, while body 2 remains fixed. Density $\rho_1(r)$ undergoes a change $\Delta\rho_1 \simeq -\nabla\rho_1\Delta r$ resulting in a new contact force at the starting point r:

$$f(r) = \frac{k\mathscr{V}}{2}(\bar{\rho} - \nabla\rho_1^T\Delta r)(\nabla\rho_2 - \nabla\rho_1 + \frac{\partial^2\rho_1}{\partial r^2}\Delta r)$$

The first-order variation of force is thus:

$$\Delta f(r) \simeq \frac{k\mathscr{V}}{2}\left(-\nabla\rho_1^T\Delta r(\nabla\rho_2 - \nabla\rho_1) + \bar{\rho}\frac{\partial^2\rho_1}{\partial r^2}\Delta r\right)$$

and we can write:

$$\frac{\Delta f(r)}{\Delta r} \simeq \frac{k\mathscr{V}}{2}(-\nabla\rho_1^T(\nabla\rho_2 - \nabla\rho_1) + \bar{\rho}\frac{\partial\nabla\rho_1}{\partial r}) \tag{7}$$

At the same time, the force at the end point $r+\Delta r$ becomes:

$$f(r+\Delta r) = \frac{k\mathscr{V}}{2}(\bar{\rho} + \nabla\rho_2^T\Delta r)(\nabla\rho_2 + \frac{\partial^2\rho_2}{\partial r^2}\Delta r - \nabla\rho_1)$$

and we obtain:

$$\frac{\Delta f(r)}{\Delta r} \simeq \frac{k\mathscr{V}}{2}(\nabla\rho_2^T(\nabla\rho_2 - \nabla\rho_1) + \bar{\rho}\frac{\partial\nabla\rho_2}{\partial r}) \tag{8}$$

Considering an infinitely small Δr and averaging the variations at the starting point (eq. 7) and at the end point of the displacement (eq. 8), we obtain the symmetric stiffness matrix of equation 3. The same result is obtained by considering a small displacement of body 2 while body 1 remains fixed.

B The stiffness of bodies in contact

We write the net generalized force as a function of the independent degrees of freedom as:

$$Q(q) = Q_{internal}(q) + \frac{\partial r}{\partial q}^T f$$

where $Q_{internal}$ is the internal force and f the contact force. This can be computed in three steps The variation of the generalized forces due to a variation of q is:

$$\Delta Q = \frac{\partial Q_{internal}}{\partial q}\Delta q + \frac{\partial r}{\partial q}^T\frac{\partial f}{\partial r}\frac{\partial r}{\partial q}\Delta q + \nabla_q\frac{\partial r}{\partial q}^T(f)$$

where $\nabla_q\frac{\partial r}{\partial q}^T$ is a function of the geometric properties of the mapping $r(q)$ and computed by the mapping. Given Δq, the first term is computed internally by the bodies. The second term is computed in three steps, by first propagating Δq to the Eulerian layer through the mapping, then applying contact stiffness, and finally mapping the result back to the bodies. The last term is computed by the mappings.

References

[Bar93] BARAFF D.: Non-penetrating rigid body simulation. In *State of the Art Reports, Eurographics* (1993).

[Bar94] BARAFF D.: Fast contact force computation for nonpenetrating rigid bodies. In *SIGGRAPH* (1994).

[Ben92] BENSON D.: Computational methods in lagrangian and eulerian hydrocodes. *Comput. Meth. in Appl. Mech. and Eng. 99* (1992).

[BFA02] BRIDSON R., FEDKIW R., ANDERSON J.: Robust treatment of collisions, contact and friction for cloth animation. In *SIGGRAPH* (2002).

[BGOS06] BARGTEIL A. W., GOKTEKIN T. G., O'BRIEN J. F., STRAIN J. A.: A semi-lagrangian contouring method for fluid simulation. *ACM Trans. Graph. 25*, 1 (2006).

[BO04] BRADSHAW G., O'SULLIVAN C.: Adaptive medial-axis aproximation for sphere-tree construction. *ACM Transactions on Graphics 23*, 1 (2004).

[BW98] BARAFF D., WITKIN A.: Large steps in cloth simulation. In *SIGGRAPH* (1998).

[CGFO04] CHENTANEZ N., GOKTEKIN T. G., FELDMAN B., O'BRIEN J. F.: Simultaneous coupling of fluids and deformable bodies. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2004).

[CLMP95] COHEN J., LIN M., MANOCHA D., PONAMGI M.: I-COLLIDE: An interactive and exact collision detection system for large-scale environments. In *Symposium on Interactive 3D Graphics* (1995).

[CMT04] CARLSON M., MUCHA P. J., TURK G.: Rigid fluid: animating the interplay between rigid bodies and fluid. *SIGGRAPH* (2004).

[ELF05] ENRIGHT D., LOSASSO F., FEDKIW R.: A fast and accurate semi-lagrangian particle level set method. *Computers and Structures 83* (2005).

[EMF02] ENRIGHT D., MARSCHNER S., FEDKIW R.: Animation and rendering of complex water surfaces. In *SIGGRAPH* (2002).

[FF01] FOSTER N., FEDKIW R.: Practical animation of liquids. In *SIGGRAPH* (2001).

[FL01] FISHER S., LIN M. C.: Fast penetration depth estimation for elastic bodies using deformed distance fields. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2001)* (2001).

[FM96] FOSTER N., METAXAS D.: Realistic animation of liquids. *Graph. Models Image Process. 58*, 5 (1996).

[FSG03] FUHRMANN A., SOBOTTKA G., GROSS C.: Distance fields for rapid collision detection in physically based modeling. In *Graphicon* (2003).

[FUF06] FUNFZIG C., ULLRICH T., FELLNER D.: Hierarchical spherical distance fields for collision detection. *IEEE Computer Graphics and Applications* (2006).

[GASF94] GARCIA-ALONSO A., SERRANO N., FLAQUER J.: Solving the collision detection problem. *IEEE Computer Graphics and Applications 13*, 3 (1994).

[GBF03] GUENDELMAN E., BRIDSON R., FEDKIW R.: Nonconvex rigid bodies with stacking. *SIGGRAPH* (2003).

[GHD03] GÉNEVAUX O., HABIBI A., DISCHLER J.-M.: Simulating fluid-solid interaction. In *Graphics Interface* (2003).

[GJK88] GILBERT E., JOHNSON D., KEERTHI S.: A fast procedure for computing the distance between objects in three-dimensional space. *IEEE J. Robotics and Automation RA*, 4 (1988).

[GLM96] GOTTSCHALK S., LIN M. C., MANOCHA D.: Obbtree: a hierarchical structure for rapid interference detection. In *SIGGRAPH* (1996).

[GSLF05] GUENDELMAN E., SELLE A., LOSASSO F., FEDKIW R.: Coupling water and smoke to thin deformable and rigid shells. In *SIGGRAPH* (2005).

[Hah88] HAHN J. K.: Realistic animation of rigid bodies. In *SIGGRAPH* (1988).

[HK05] HONG J.-M., KIM C.-H.: Discontinuous fluids. *ACM Trans. Graph. 24*, 3 (2005).

[HTG03] HEIDELBERGER B., TESCHNER M., GROSS M.: Real-time volumetric intersections of deforming objects. In *Proceedings of Vision, Modeling, Visualization (VMV)* (2003).

[JP04] JAMES D. L., PAI D. K.: Bd-tree: output-sensitive collision detection for reduced deformable models. In *SIGGRAPH* (2004).

[JTT01] JIMENEZ P., THOMAS F., TORRAS C.: 3d collision detection: A survey. *Computer and Graphics 21*, 3 (2001).

[KFCO06] KLINGNER B. M., FELDMAN B. E., CHENTANEZ N., O'BRIEN J. F.: Fluid animation with dynamic meshes. In *SIGGRAPH* (2006).

[KHM*98] KLOSOWSKI J. T., HELD M., MITCHELL J. S. B., SOWIZRAL H., ZIKAN K.: Efficient collision detection using bounding volume hierarchies of k-dops. *IEEE Transactions on Visualization and Computer Graphics 4*, 1 (1998).

[LAM01] LARSSON T., AKENINE-MOLLER T.: Collision detection for continuously deforming bodies, 2001.

[LG98] LIN M. C., GOTTSCHALK S.: Collision detection between geometric models: a survey. In *Proc. of IMA Conference on Mathematics of Surfaces* (1998), pp. 37–56.

[LGF04] LOSASSO F., GIBOU F., FEDKIW R.: Simulating water and smoke with an octree data structure. In *SIGGRAPH* (2004).

[LSSF06] LOSASSO F., SHINAR T., SELLE A., FEDKIW R.: Multiple interacting liquids. In *SIGGRAPH* (2006).

[MAC04] MARCHAL D., AUBERT F., CHAILLOU C.: Collision between deformable objects using fast-marching on tetrahedral models. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2004).

[MC95] MIRTICH B., CANNY J.: Impulse-based simulation of rigid bodies. In *Interactive 3D graphics* (1995).

[MCG03] MÜLLER M., CHARYPAR D., GROSS M.: Particle-based fluid simulation for interactive applications. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2003).

[MKE03] MEZGER J., KIMMERLE S., ETZMUSS O.: Hierarchical Techniques in Collision Detection for Cloth Animation. *Journal of WSCG 11*, 2 (2003).

[MST*04] MÜLLER M., SCHIRM S., TESCHNER M., HEIDELBERGER B., GROSS M.: Interaction of fluids with deformable solids. *J. of Computer Animation and Virtual Worlds (CAVW) 15*, 3-4 (2004).

[MW88] MOORE M., WILHELMS J.: Collision detection and response for computer animation. In *SIGGRAPH* (1988).

[Pro97] PROVOT X.: Collision and self-collision handling in cloth model dedicated to design garments. In *Graphics Interface* (1997).

[PTB*03] PREMOŽE S., TASDIZEN T., BIGLER J., LEFOHN A., WHITAKER R. T.: Particle-based viscoelastic fluid simulation. *Eurographics* (2003).

[SAC*99] STORA D., AGLIATI P.-O., CANI M.-P., NEYRET F., GASCUEL J.-D.: Animating lava flows. In *Graphics Interface* (1999).

[SGG*06] SUD A., GOVINDARAJU N., GAYLE R., KABUL I., MANOCHA D.: Fast proximity computation among deformable models using discrete voronoi diagrams. In *SIGGRAPH* (2006).

[Sta99] STAM J.: Stable Fluids. In *Proceedings of ACM SIGGRAPH 99* (August 1999), pp. 121–128.

[THM*03] TESCHNER M., HEIDELBERGER B., MÜLLER M., POMERANETS D., GROSS M.: Optimized spatial hashing for collision detection of deformable objects. In *Proceedings of Vision, Modeling, Visualization (VMV)* (2003).

[TKH*04] TESCHNER M., KIMMERLE S., HEIDELBERGE B., ZACHMANN G., RAGHUPATHI L., FUHRMANN A., CANI M.-P., FAURE F., MAGNENAT-THALMANN N., STRASSER W., VOLINO P.: Collision detection for deformable objects. In *Proc. State of the Art Reports, Eurographics 04* (2004).

[vdB97] VAN DEN BERGEN G.: Efficient collision detection of complex deformable models using aabb trees. *J. Graph. Tools 2*, 4 (1997).

[VMT95] VOLINO P., MAGNENAT-THALMANN N.: Collision and self-collision detection: Efficient and robust solutions for higly deformable surfaces. *Comp. Animation and Simulation* (1995).

[VMT00] VOLINO P., MAGNENAT-THALMANN N.: Implementing fast cloth simulation with collision response. In *Computer Graphics International* (2000).

[VMT06] VOLINO P., MAGNENAT-THALMANN N.: Resolving surface collisions through intersection contour minimization. In *SIGGRAPH* (2006).

[ZL03] ZACHMANN G., LANGETEPE E.: Geometric data structures for computer graphics. In *SIGGRAPH Tutorial* (2003).

Unité de recherche INRIA Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399